

# Estructuras de Datos 1 - ST0245

## Examen Parcial 1 - 032 - Martes

Nombre:.....  
Departamento de Informática y Sistemas  
Universidad EAFIT

Septiembre 05 de 2017

### Criterios de calificación

- Selección múltiple con única respuesta
  - Respuesta correcta: 100 %
  - Respuesta incorrecta: 0 %
- Completar código
  - Respuesta correcta 100 %
  - Respuesta incorrecta o vacía 0 %

#### NOTAS IMPORTANTES:

- Responda en la hoja de PREGUNTAS
- Marque la hoja de PREGUNTAS

### 1. Notación O grande 10 %

Después de hacer un análisis de complejidad de un algoritmo  $A(n)$ , obtuvimos que ejecuta, en el peor caso pasos,  $T(n) = 2^n + n^2 + 7n$ . ¿Cuál es el orden asintótico del algoritmo?

O(-----)

### 2. Listas 10 %

En algoritmiandia existe un método `busqueda` que busca un número en un arreglo de números, con una complejidad de  $O(\log n)$ , donde  $n$  es el tamaño del arreglo. Existe otro método, llamado `metodo1`, que tiene dos ciclos anidados y dentro del ciclo interno se llama al método `busqueda` con cada uno de los elementos de una matriz. Si un elemento de la matriz, es

decir `matriz[i][j]`, se encuentra en el arreglo `sec`, el elemento se añade al final de una lista doblemente enlazada (`LinkedList`).

```
public List<Integer> metodo1(int[] [] matriz, int[] sec){
    LinkedList<Integer> lista = new LinkedList<>();
    for(int i = 0; i < matriz.length; ++i){
        for(int j = 0; j < matriz[0].length; ++j){
            boolean esta = busqueda(sec, matriz[i][j]);
            if(esta)
                lista.add(matriz[i][j]);
        }
    }
    return lista;
}
```

¿Cuál es la complejidad asintótica del método `metodo1` asumiendo que la matriz es de  $n \times m$  y el arreglo `sec` es de tamaño  $n$ ?

O(-----)

### 3. Listas 10 %

¿Cuál operación tiene una mayor complejidad asintótica, para el peor de los casos, en un lista simplemente enlazada?

- A Insertar un elemento en la mitad de la lista
- B Borrar el elemento que está en la mitad de la lista
- C Obtener el elemento que está en la mitad de la lista
- D Las tres tienen la misma complejidad asintótica

## 4. Complejidad 10 %

El siguiente algoritmo calcula la suma de los elementos de una matriz cuadrada de tamaño  $n$ .

```
public int suma(int[] [] m) {  
    int sum = 0;  
    for(int i=0; i < m.length; i++){  
        for(int j=0; j < m.length; j++){  
            sum = sum + m[i][j];  
        }  
    }  
    return sum;  
}
```

¿Cuál es su complejidad asintótica en el peor de los casos?

O(\_\_\_\_\_)

## 5. Complejidad 20 %

El siguiente algoritmo cuenta del  $n$  al 1.

```
public void imprimir(int n) {  
    if (n == 1) println(1);  
    else { println(n);  
          imprimir(n-1); }  
}
```

(10 %) ¿Cuál es el número de pasos que ejecuta para el peor de los casos?

$T(n) =$  \_\_\_\_\_

(10 %) ¿Cuál es la complejidad asintótica en el peor de los casos?

O(\_\_\_\_\_)

## 6. Recursion 40 %

Hay un tablero de  $2 \times n$  cuadrados y usted necesita saber de cuantas maneras se puede llenar el tablero usando rectángulos de  $1 \times 2$ . Se ha propuesto el siguiente algoritmo recursivo

```
1 public int formas(int n){  
2     if(n <= 2) _____;  
3     return formas(____) +  
4         formas(____);  
5 }
```

Complete las líneas faltantes.

(10 %) Línea 2. \_\_\_\_\_

(10 %) Línea 3. \_\_\_\_\_

(10 %) Línea 4. \_\_\_\_\_

(10 %) ¿Cuántas instrucciones ejecuta el algoritmo en el peor de los casos?

- A.  $T(n) = T(n-1) + C$
- B.  $T(n) = T(n-1) + T(n-2) + C$
- C.  $T(n) = T(n/2) + C$
- D.  $T(n) = T(n+1) + C$

**Pista:** Considere llenar un tablero de  $2 \times n$ . Si le quitamos la primera baldosa tenemos un tablero de  $2 \times (n-1)$  (usando recursión). Si le quitamos 2 baldosas queda un tablero de  $2 \times (n-2)$ . Haga el dibujo. ¿Se le pueden quitar 3 baldosas, o con lo anterior ya puedo formar el de  $2 \times (n-3)$ ?